UNIVERSITÉ DE MONTRÉAL

CENTRE DE RECHERCHE SUR LES TRANSPORTS

CASE POSTALE 6128, SUCCURSALE A, MONTRÉAL, QUÉBEC, H3C 3J7

THE SINGLE DEPOT VEHICLE SCHEDULING PROBLEM

by

Jacques A. Ferland

Université de Montréal
Centre de recherche sur les transports - Publication #385
Février 1985

Dépôt légal - 2e trimestre 1985

ABSTRACT

The vehicle scheduling problem is specified in terms of a set of tasks to be executed with a fleet of vehicles. The purpose of this paper is to specify the one depot problem, to introduce several approaches, but not to report any numerical comparison of these. The problem with fixed start time is first analyzed. Then, heuristic and exact approaches for the problem with time windows constraints are reviewed.

RÉSUMÉ

Le problème d'horaires de véhicules est défini en termes d'un ensemble de tâches qui doivent être complétées par une flotte de véhicules. Le problème est d'abord défini explicitement puis, plusieurs approches pour aborder le problème sont présentées, sans pour autant les comparer numériquement. Nous analysons d'abord le problème où les moments pour entreprendre les tâches sont fixes. Ensuite, nous présentons plusieurs approches heuristiques et exactes pour traiter le problème avec fenêtres de temps.

1. INTRODUCTION

The vehicle scheduling problem can be specified in terms of a set of tasks to be executed with a fleet of vehicles. Each task i includes time and space (location) characteristics specified as follows :

- a start location SL_i
- an end location EL_i
- a start time ST_i
- a time duration D_i
- an end time $ET_i = ST_i + D_i$.

To execute task i a vehicle must be at location SL_i at time ST_i. Then at time ET_i the vehicle is located at EL_i and becomes available for other tasks. The problem is to partition the set of tasks into sequences of tasks, with each sequence corresponding to the schedule of a vehicle. Hence no two tasks of a sequence can take place simultaneously. Furthermore task j can be executed after task i in the sequence if, taking into account the time t_{ij} required for a vehicle to go from EL_i to SL_j, the vehicle can be made available at time ST_j at SL_j; i.e.

$$ET_i + t_{ij} \leq SL_j.$$

The fleet of vehicles available can be characterized by its size (the number of vehicles), by the locations of the depots where the vehicles start and end their schedules, and by the types of vehicles, etc. In this paper we are concerned with the problem in which there is a single depot and all vehicles are of the same type.

The objective function to be minimized can be specified in terms of operating costs, or in terms of investment costs, or in terms of both. For instance, the objective might be to reduce the fleet size to a minimum.

The vehicle scheduling problem is presented and analyzed in Bodin et al. (1983) where procedures for handling different variants are summarised. The purpose of this paper is to complete this survey for the variant with a single depot. In particular, many authors have recently addressed the problem of vehicle scheduling with time window constraints where the start time ST_i of each task i must lie within some time interval. The purpose of this paper is to summarize the procedures rather than to analyze and comment on their relative efficiency.

In Section 2 we analyze the case where the start time ST_i of each task i is fixed. Then in Section 3 we turn to the vehicle scheduling problem with time window constraints. Heuristic and exact (branch-and-bound) approaches are reviewed in Sections 4 and 5, respectively.

2. FIXED START TIME

A mathematical formulation is derived for this problem using a precedence graph of the tasks, (N,A), characterized as follows :

i) With each task i associate a node i in N. Two nodes, s and t, are associated with the depot to indicate where the vehicles start and finish.

ii) An arc (i,j) exists (i.e., $(i,j) \in A$, or task j can follow task i) if

$$ET_i + t_{ij} < ST_j.$$

Furthermore, for each task i, introduce arcs (s,i) and (i,t) allowing i to be the first or the last task of a sequence, respectively.

This graph is acyclic as illustrated in Figure 1.

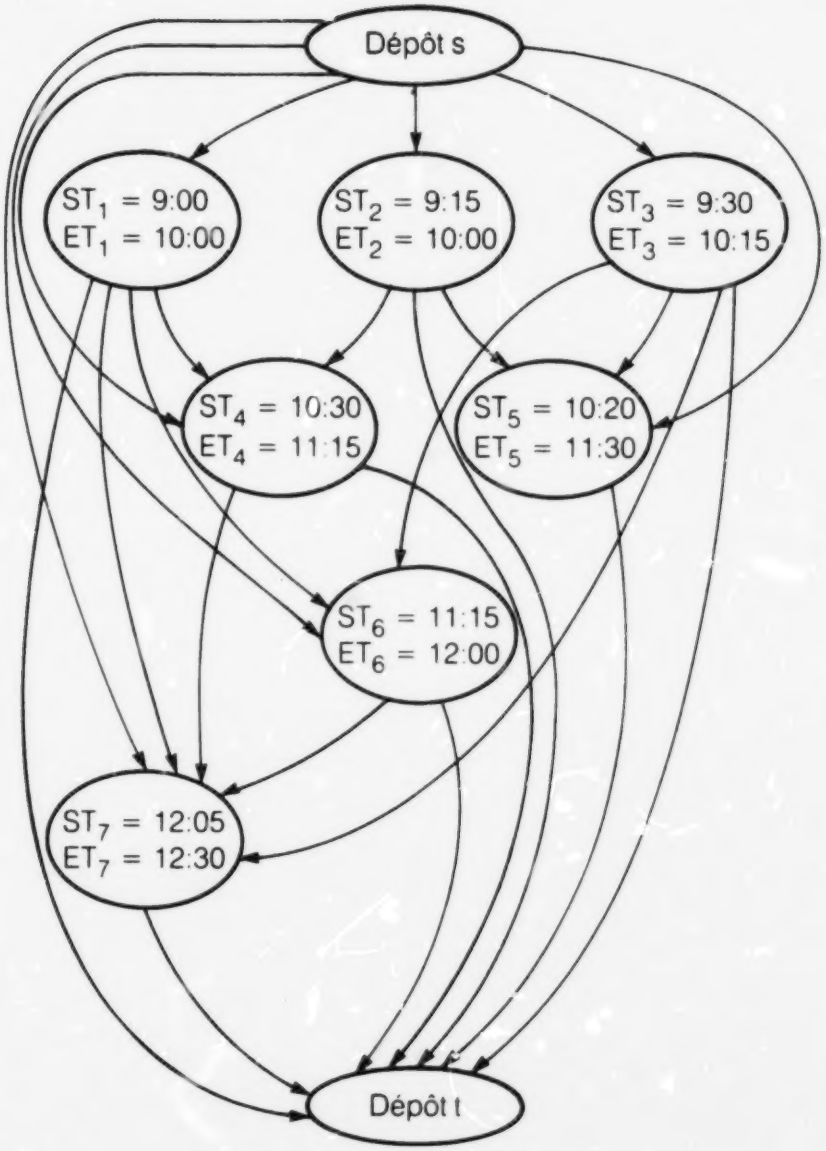

Dépôt s
ST1 = 9:00
ET1 = 10:00
ST2 = 9:15
ET2 = 10:00
ST3 = 9:30
ET3 = 10:15
ST4 = 10:30
ET4 = 11:15
ST5 = 10:20
ET5 = 11:30
ST6 = 11:15
ET6 = 12:00
ST7 = 12:05
ET7 = 12:30
Dépôt t

FIGURE 1

In terms of this precedence graph, the problem is to determine a partition of the nodes into a set of paths (or sequences), each of these starting at s and ending at t, to minimize an objective function.

The problem can be formulated as a minimum cost flow problem defined on the precedence graph as follows :

<u>Problem (P)</u> Min $\sum_{(i,j)\in A} c_{ij} x_{ij}$

Subject to $\sum_{i\in N} x_{ij} = 1 \qquad j\in N-\{s,t\}$

$\sum_{j\in N} x_{ij} = 1 \qquad i\in N-\{s,t\}$

$x_{ij} = 0 \text{ or } 1 \qquad (i,j)\in A$

where c_{ij} is the cost of traversing arc (i,j) and the decision variables

$$x_{ij} = \begin{cases} 1 & \text{if some vehicle traverses arc } (i,j) \\ 0 & \text{otherwise.} \end{cases}$$

The linear costs c_{ij} are specified according to the objective of the problem. For instance, if the objective is to minimize fleet size, then

$$c_{ij} = \begin{cases} 1 & \text{if } i = s \\ 0 & \text{if } i \neq s, \end{cases}$$

and if the objective is to minimize fleet size and deadheading time, then

$$c_{ij} = \begin{cases} t_{ij} & \text{if } i \text{ and } j \text{ are different from } s,t \\ K & \text{if } i = s \\ 0 & \text{if } j = s \end{cases}$$

where the magnitude of K relative to the t_{ij} measures the importance of fleet size versus deadheading time.

In general, this problem can be solved using a minimum cost flow algorithm. But if the objective is to minimize fleet size, then it reduces to Dilworth's chain decomposition problem and it can be solved with a variant of the maximal flow labeling procedure (see Ford and Fulkerson, 1962). Furthermore, if the start and end locations of all tasks are located in a few cities (i.e., the same city is the start or end location of several tasks) the following procedure due to Saha (1970) can be used : for each city, order chronologically the arrivals and departures from it, and assign arriving vehicles to departures according to a first-in-first-out strategy; when no vehicle is available for a departure, take a new vehicle out of the depot.

3. TIME WINDOWS

In Section 2 it was assumed that ST_i is fixed for all tasks i. Now suppose that for each task i an earliest starting time a_i and a latest starting time b_i are specified :

$$a_i \leq ST_i \leq b_i.$$

The time interval $[a_i,b_i]$ is called the <u>time window</u> associated with task i.

A mathematical formulation is derived for this problem using the following oriented graph (N,A) :

i) With each task i associate a node i in N. With the depot, associate two nodes s and t.

ii) An arc (i,j) exists (i.e., $(i,j) \in A$ or task j might follow task i) if

$$a_i + D_i + t_{ij} \leq b_j.$$

Furthermore, for each task i, arcs (s,i) and (i,t) exist.

In terms of this oriented graph, the problem is to partition the nodes into a set of paths (or sequences), each of these starting at s and ending at t, to minimize an objective function such that the time window constraints are satisfied for each task. It is summarized as follows :

<u>Problem (PW)</u> Min $\sum_{(i,j)\in A} c_{ij} x_{ij}$

Subject to

$$\sum_{i\in N} x_{ij} = 1 \qquad j\in N-\{s,t\}$$

$$\sum_{j\in N} x_{ij} = 1 \qquad i\in N-\{s,t\}$$

$$x_{ij} = 0 \text{ or } 1 \qquad (i,j)\in A$$

$$x_{ij} > 0 \Rightarrow ET_i + t_{ij} \leq ST_j \qquad (i,j)\in \overline{A} \qquad (3.1)$$

$$a_i \leq ST_i \leq b_i \qquad i\in N-\{s,t\} \qquad (3.2)$$

where $\overline{A} = \{(i,j)\in A : i \neq s,t, \quad j \neq s,t\}$, c_{ij} is the cost of traversing arc (i,j) and the decision variables

$$x_{ij} = \begin{cases} 1 & \text{if some vehicle traverses arc } (i,j) \\ 0 & \text{otherwise.} \end{cases}$$

Constraints (3.1) and (3.2) are referred to as the time window constraints.

Several methods have been proposed to solve this problem. Some of these are heuristic methods generating "good" solutions and are reviewed in Section 4. Other methods generate optimal solutions using "branch-and-bound" procedures. Some of these are reviewed in Section 5.

4. HEURISTIC METHODS

Four different heuristic methods are now summarized. The first method, proposed by Levin (1971) and by Swersey and Ballard (1984), solves

a discrete approximation of problem (PW) specified in Section 3. The second method due to Orloff (1976) is a two phase procedure : phase 1 generates a feasible solution and phase 2 improves on it. The third method, proposed by Bodin and Berman (1979) and by Desrosiers et al. (1980), finds a feasible solution by solving a sequence of transportation problems. The fourth heuristic, presented in Graham and Nuttle (1984), generates a feasible solution using problem (PW) where the time window constraints are relaxed (or deleted).

4.1 Discrete approximation

The time window interval of each task is discretized in the sense that specific moments in the time interval are selected as candidate start times for the task. To derive the model, replace each variable ST_i in Problem (PW) by a set of binary decision variables y_ν associated with candidate start times for task i. With these variables we specify a mathematical model very similar to Problem (P) in Section 2. Indeed consider the following precedence graph $(\tilde{N},\tilde{A})$:

i) With each of the specific candidate starting times of each task, associate a node ν. s and t are nodes associated with the depot.

ii) An arc (τ,ν) exists if a vehicle can execute task j starting at the specific time ν after task i starting at specific time τ. Also, for each $\nu\in\tilde{N}-\{s,t\}$, the arcs (s,ν) and (ν,t) exist.

Denote by $\Gamma_i = \{\nu\in N : \nu \text{ is candidate start time of task } i\}$.

The mathematical model is summarized as follows :

$$\text{Min} \quad \sum_{(\tau,\nu)} c_{\tau\nu} x_{\tau\nu}$$

$$\text{Subject to} \quad \sum_{\tau \in \tilde{N}} x_{\tau\nu} = y_\nu \qquad \nu \in \tilde{N} - \{s,t\}$$

$$\sum_{\tau \in \tilde{N}} x_{\nu\tau} = y_\nu \qquad \nu \in \tilde{N} - \{s,t\}$$

$$x_{\tau\nu} = 0 \text{ or } 1 \qquad (\tau,\nu) \in \tilde{A}$$

$$\sum_{\nu \in \Gamma_i} y_\nu = 1 \qquad 1 \leq i \leq m$$

$$y_\nu = 0 \text{ or } 1 \qquad \nu \in \tilde{N} - \{s,t\}$$

where $c_{\tau\nu}$ is the cost of traversing arc (τ,ν) and the decision variables

$$x_{\tau\nu} = \begin{cases} 1 & \text{if a vehicle traverses arc } (\tau,\nu) \\ 0 & \text{otherwise.} \end{cases}$$

Swersey and Ballard (1984) have used this model in the context of school busing where the objective is to reduce the fleet size. In this paper, they report that the optimal solution for the linear programming relaxation of the model where constraints $x_{\nu\tau} = 0$ or 1 are replaced by $0 \leq x_{\nu\tau} \leq 1$ and constraints $y_\nu = 0$ or 1 are replaced by $0 \leq y_\nu \leq 1$, was integer in many of their examples.

4.2 Matching methods

To specify the two phase method due to Orloff (1976), consider the following oriented graph :

i) With each task i associate a node i.

ii) With each pair of nodes i and j, associate two arcs :

. arc (i,j) with cost

$$c_{ij} = \begin{cases} t_{ij} & \text{if } a_i + D_i + t_{ij} \leq b_j \\ K & \text{otherwise} \end{cases}$$

. arc (j,i) with cost

$$c_{ij} = \begin{cases} t_{ji} & \text{if } a_j + D_j + t_{ji} \leq b_i \\ K & \text{otherwise.} \end{cases}$$

During the first iteration of phase 1, a minimum weighted matching problem is solved on the preceding graph. This is essentially equivalent to identifying a set of arcs such that each node is the extremity of one and only one arc (i.e., no two arcs have a node in common) and the sum of their cost is minimized. If an arc (i,j) belongs to this set, and if $c_{ij} < K$, then collapse nodes i and j into one node i' and specify accordingly a time window interval $[a_{i'}, b_{i'}]$ for this pair of tasks assumed to be collapsed into one. Then specify a new oriented graph with some of the tasks collapsed into one, and repeat the procedure.

When no further matchings are possible (i.e., when the optimal solution of the minimum weighted matching problem includes only arcs with cost K), each node of the current graph corresponds to the schedule of a vehicle executing the sequence of tasks collapsed at this node. This solution is feasible for problem (PW), but is not optimal in general. Assume that z vehicles are necessary for this solution.

During phase 2, the solution generated in phase 1 is improved. Consider the oriented graph (N,A) specified in Section 3, but modify it to include z nodes associated with the depot, each with arcs to and from every task node. Also, add arcs to and from every depot node and every other depot node with cost 0. The solution generated in phase 1 corresponds to a travelling salesman tour in this augmented graph if we assume that each depot node is the starting node of one and only one vehicle and the ending node of one and only one other vehicle. The exchange procedure proposed in Lin (1965) can be adapted to improve this initial tour.

This procedure always yields a feasible solution which is not, in general, optimal.

4.3 Assignment method

To specify the method, we refer to the oriented graph (N,A) introduced in Section 3, but the depot nodes will not be relevant. Denote $N' = N-\{s,t\}$.

At each iteration of the procedure, a transportation problem is specified in terms of two subsets of tasks, N_1 and N_2. The subset N_1 is the set of source-tasks including the tasks already executed by a vehicle which is now available for other tasks. The subset N_2 includes a subset of sink-tasks to be executed (i.e., for which a vehicle is required).

For the first iteration, the subset N_1 includes the tasks in N' that cannot be executed after any other task. Hence $j \in N_1$ if $a_i + D_i + t_{ij} > b_j$ for all $i \in N'$, $i \neq j$. Also, if two tasks can follow each other but cannot be executed after any other task, then select one of these and include it in N_1. Hence each task in N_1 should be the first task of a different sequence. Furthermore, once a task in N_1 is executed, the vehicle is available for other tasks. The sequences are built up by solving a sequence of transportation problems. To determine the subset N_2 of sink-tasks to be executed next, replace N' by $N'-N_1$, and repeat the preceding procedure used to define N_1.

For each pair of nodes $i \in N_1$, $j \in N_2$, define an asignment cost ac_{ij} for using a vehicle to execute j after i as follows :

$$ac_{ij} = \begin{cases} t_{ij} & \text{if } a_i + D_i + t_{ij} < b_j \\ K_1 & \text{otherwise} \end{cases}$$

where $K_1 >> K$. Introduce a dummy source $\bar{s}$ and a dummy sink $\bar{t}$. For all $i \in N_1$ define an assignment cost $ac_{i\bar{t}} = 0$ to indicate that the vehicle becomes available after executing task i. For all $j \in N_2$ define an assignment cost $ac_{\bar{s}j} = K$ to indicate that a new sequence is initiated with task j.

This transportation problem is then solved. If $i \in N_1$ is assigned to $\bar{t}$, then i remains in set N_1. If $\bar{s}$ is assigned to $j \in N_2$, then a new sequence is initiated and the vehicle is available after executing task j, and j is introduced into N_1. If $i \in N_1$ is assigned to $j \in N_2$, then the vehicle executes j after executing i (i.e. i and j belong to the same sequence). Then i is eliminated from N_1, j is introduced into N_1 because the vehicle is no longer available after task i but rather after task j. Furthermore the earliest time $a_j + D_j$ at which the vehicle is available is adjusted accordingly to the sequence to which it belongs. Also we make the assumption that each task of a sequence is executed as early as possible.

The procedure is repeated. The current set N_1 of source-tasks is used. The set N_2 is determined using the same procedure with N' replaced by the set of node-tasks that have not been assigned to any sequence as yet. The process stops when this set is empty.

The method yields a feasible solution which is not optimal in general.

4.4 Relaxation method

To use this method, Graham and Nuttle (1984) make the assumption that the oriented graph (N,A) is acyclic. This assumption is acceptable in most applications where task durations are longer than the length of the time window interval.

First, problem (PW) where the time window constraints (3.1) and (3.2) are relaxed (deleted) is solved. Note that the relaxed problem reduces to an assignment problem. The optimal solution generates a partition of the nodes into a set of paths (sequences). Under the assumption that the graph is acyclic, each path starts at s and ends at t. But some of these sequences might be infeasible in the sense that the time window constraint is violated for at least one task of the sequence. If all sequences are feasible, the solution is optimal. Otherwise, the sequences have to be modified to recover feasibility. In their implementation of the method, the adjustments of the sequences are carried out manually by experienced scheduling personnel.

5. EXACT METHODS

In this secion we summarize branch-and-bound exact methods to find optimal solutions to problem (PW). The first method presented by Desrosiers et al. (1984) uses a column generation technique to solve a set partitioning problem equivalent to problem (PW). The second method requires the resolution of a problem of type (PW) where the time window constraints are relaxed at each node of the branch-and-bound tree until an optimal solution is reached. It is presented in Desrosiers et al. (1983a).

5.1 Column generation technique

Assume that Ω is a set of sequences of tasks, and that each of these sequences satisfies the time window constraints. Use the decision variables y_r, $r \in \Omega$ to specify the following set partitioning problem :

Problem (SP) Min $\sum_{r \in \Omega} c_r y_r$

Subject to $\sum_{r \in \Omega} \delta_{ri} y_r = 1 \qquad 1 \leq i \leq n$

$y_r = 0 \text{ or } 1 \qquad r \in \Omega$

where

$$y_r = \begin{cases} 1 & \text{if sequence } r \in \Omega \text{ is used} \\ 0 & \text{otherwise.} \end{cases}$$

$$\delta_{ri} = \begin{cases} 1 & \text{if task } i \text{ belongs to sequence } r \\ 0 & \text{otherwise,} \end{cases}$$

and c_r is defined according to the c_{ij} and is the cost for a vehicle to execute the r^{th} sequence of tasks.

At each iteration of the procedure the linear programming relaxation of a problem (SP) associated with a given Ω is solved. This relaxation is obtained by replacing the constraints $y_r = 0$ or 1 by $y_r \geq 0$. Denote by σ_i the optimal multiplier associated with constraint $\sum_{r \in \Omega} \delta_{ri}\, y_r = 1$. Pricing out the r^{th} sequence, we obtain the relative cost

$$\sum_{(i,j) \in A} \delta_{ri}\, \delta_{rj}\, c_{ij} - \sum_{i \in N} \delta_{ri}\, \sigma_i .$$

Hence the approach is to generate new sequences not already in Ω which might reduce the overall cost of the set partitioning problem. The new sequences have negative relative costs when evaluated with the optimal multipliers σ_i. It is easy to see that such a sequence is the optimal solution to a shortest path problem with the additional time window constraints in the oriented graph (N,A) when the cost of arc (i,j) is taken equal to $(c_{ij} - \sigma_i)$. Desrosiers et al. (1983b) propose an algorithm to solve this problem.

This procedure is embedded into a branch-and-bound method to obtain an optimal integer solution to a problem (SP) equivalent to (PW). At a node of the branch-and-bound tree, if the linear programming relaxation of the set partitioning problem has a fractional optimal solution, then branching

takes place on the sequence for which the associated decision variable can be set to 1 at the least additional cost according to the estimates given by the marginal costs. Furthermore, the branching is completed according to the Bellmore and Malone (1971) method as adapted by Carpaneto and Toth (1980) for the travelling salesman problem. This is illustrated in Figure 2 where we assume that sequence r includes tasks i_1, i_2, ..., i_m. The first branch imposes the existence of sequence r. The second branch imposes the sequence i_1, i_2, ..., i_{m-1} but prohibits that i_m follows i_{m-1}, and so on. The next to last branch imposes the sequence i_1, i_2 but prohibits that i_3 follows i_2. The last branch prohibits that i_2 follows i_1.

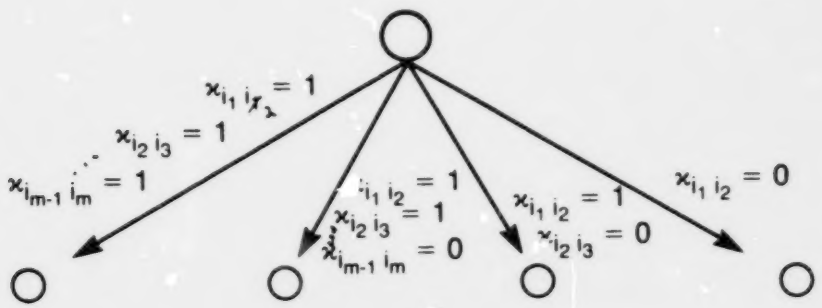


FIGURE 2

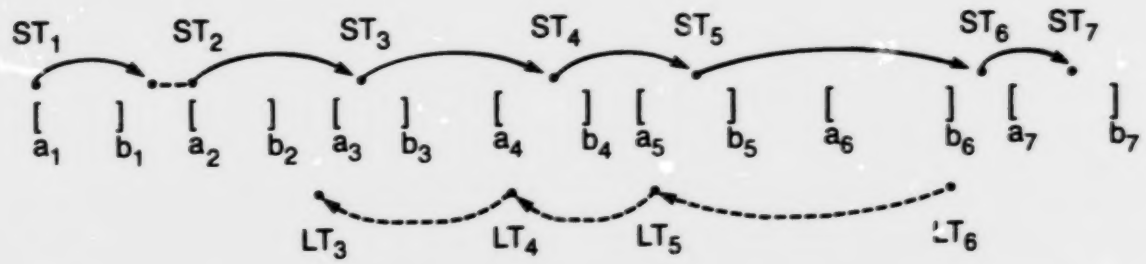
5.2 Time-window constraints relaxation

As in Section 4.4, consider problem (PW) where the time window constraints are relaxed (i.e., discarded). Sequences of tasks can be generated by solving this relaxed version of (PW). If the assumption that the graph (N,A) is acyclic is not verified, this optimal solution might include subtours where the vehicle can return to the first task of the sequence after executing the last task of the sequence. Furthermore, the time window constraints might not be satisfied for some tasks in the sequences generated. Of course, if neither of these cases occurs, then the solution is optimal for (PW).

On the other hand, to eliminate subtours and sequences where some time window constraints are not satisfied, a branch-and-bound procedure is used where a relaxed version of a problem of type (PW) is solved at each node of the branch-and-bound tree.

Two branching procedures are presented by Desrosiers et al. (1983a). They are applied only on the infeasible part of the sequence including the tasks for which the time window constraints of the last task of the subsequence cannot be satisfied even in the first task of the subsequence is executed as early as possible.

This notion of infeasible subsequence is illustrated in Figure 3 where the subsequence of tasks 3, 4, 5 and 6 is not feasible because the time-window constraint of task 6 is violated even if 3 is initiated at a_3. Indeed, to be able to initiate task 6 at b_6 (latest starting time for 6), task 3 should be initiated no later than LT_3 which is earlier than a_3.



ST_i is the starting time of task i in the sequence.
LT_i is the latest starting time of task i according to the sequence.

FIGURE 3

As in Section 5.1, the branching procedures proposed by Desrosiers et al. (1983a) represent an adaptation of the Bellmore and Malone (1971) approach. The first procedure of branching on the flow variables is very similar to the procedure used in Section 5.1. It is illustrated in Figure 4

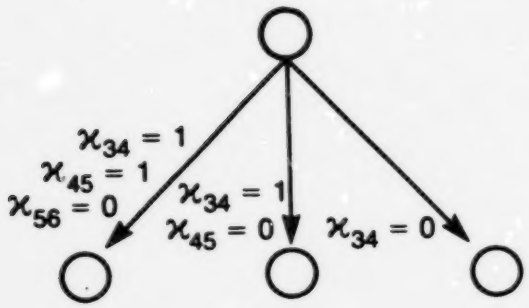


FIGURE 4

to eliminate the infeasible subsequence of tasks 3, 4, 5 and 6 of the preceding example. The first branch imposes subsequence 3, 4, 5 and prohibits the link 5,6. The second one imposes subsequence 3, 4 and prohibits link 4,5. Finally, the last branch prohibits link 3,4.

The second branching procedure is specified in term of the time subintervals of the tasks of the infeasible subsequence. These are specified according to the notation introduced in Figure 3. For each task i of the infeasible subsequence, denote

- L_i the subinterval $[a_i, LT_i]$
- R_i the subinterval $(LT_i, b_i]$.

The branching procedure illustrated in Figure 5 generates branches where some tasks are required to be initiated within one or the other subinterval. For the preceding example it is easy to verify that links (3,4), (4,5) and (5,6) are prohibited by the first, the second and the third branch, respectively.

Minor adjustments are required to use these procedures to eliminate subtours where time-window constraints are satisfied for all tasks. For the details we refer the reader to Desrosiers et al. (1983a).

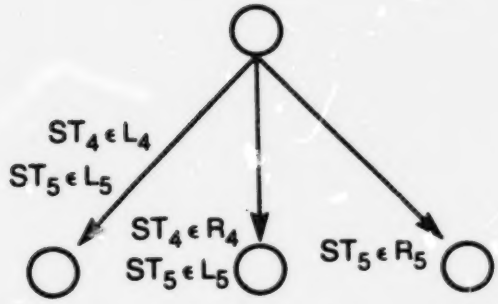


FIGURE 5

The branching strategy selecting the subtour or illegal subsequence on which branching takes place is specified in terms of a bound on the increase in the optimal value for the relaxed problem when a link is prohibited. The subtour, or illegal subsequence, generating the least number of branches where this bound is positive, is selected.

6. CONCLUSION

The purpose of this paper was to specify the one depot vehicle scheduling problem and to introduce several approaches to analyzing the problem with time window constraints. The paper does not report any numerical comparison of the relative efficiency of the methods because the author is not aware of any study of this sort, and because the use of any specific method depends on the context in which the problem arises. Of course, the exact methods are more time consuming than heuristics, but if the objective is to reduce the fleet size to a minimum, the savings of the costs of a vehicle might justify the use of exact methods.

The presentation of the different methods is very sketchy to underline only the basic principles. The reader is referred to the specific papers to see how the authors implement these basic principles and use other techniques and principles to accelerate these methods.

REFERENCES

BELLMORE, M. and MALONE, J.C. (1971). "Pathology of Travelling Salesman Subtour Elimination Algorithm", Operations Research 19, 278-307.

BODIN, L. and BERMAN, L. (1979). "Routing and Scheduling of School Buses by Computer", Transportation Research 13, 113-129.

BODIN, L., ASSAD, A., BALL, M. and GOLDEN, B. (1983). "The State of the Art in Routing and Scheduling of Vehicles and Crews", Computers and Operations Research 10, 69-211.

CARPANETO, G. and TOTH, P. (1980). "Some New Branching and Bounding Criteria for the Asymmetric Travelling Salesman Problem", Management Science 26, 736-743.

DESROSIERS, J., FERLAND, J.A., ROUSSEAU, J.-M., LAPALME, G., CHAPLEAU, L. (1980). "TRANSCOL : A Multi-Period School Bus Routing and Scheduling System", Publication #164, Centre de recherche sur les transports, Université de Montréal, Montréal, Canada; also in Ignall, E.J. and Swersey, A.J. (editors), Management Science and the Delivery of Urban Services, Elsevier-North-Holland (forthcoming).

DESROSIERS, J., SOUMIS, F. and DESROCHERS, M. (1984). "Routing with Time Windows by Column Generation", Networks 14, 545-565.

DESROSIERS, J., SOUMIS, F. and DESROCHERS, M. (1983a). "Routing and Scheduling by Network Relaxation and Branch-and-Bound on Time Variables", Publication #278, Centre de recherche sur les transports, Université de Montréal, Montréal, Canada; also in Computer Scheduling of Public Transport, Volume II, North-Holland (forthcoming).

DESROSIERS, J., PELLETIER, P. and SOUMIS, F. (1983b). "Plus court chemin avec contraintes d'horaire", RAIRO 17, 357-377.

FORD, L.R. and FULKERSON, D.R. (1962). "Flows in Networks", Princeton University Press, Princeton, New Jersey.

GRAHAM, D. and NUTTLE, L.W. (1984). "A Comparison of Heuristics for a School Bus Scheduling Problem", IE Technical Report No. 84-3, Department of Industrial Engineering, North Carolina State University, Raleigh, North Carolina.

LEVIN, A. (1971). "Scheduling and Fleet Routing Models for Transportation Systems", Transportation Science 5, 232-255.

LIN, S. (1965). "Computer Solutions of the Travelling Salesman Problem", Bell System Technical Journal 44, 2245-2269.

ORLOFF, C. (1976). "Route Constrained Fleet Scheduling", Transportation Science 10, 149-168.

SAHA, J. (1970). "An Algorithm for Bus Scheduling Problems", Operational Research Quarterly 21, 463-474.

SWERSEY, A.J. and BALLARD, W. (1984). "Scheduling School Buses", Management Science 30, 844-853.